Analysis of Numerical Methods for Solving Ordinary Differential Equations

Carl De Vries

C++ | Honors | Spring 2014 |

Numerical methods provide an alternative process for obtaining solutions to problems. Unlike analytical methods which give exact values, solutions from numerical methods are approximations. There are several types of numerical methods including methods for integrating ordinary differential equations (ODEs). Numerical methods can be useful for all ODEs, but may be required when an analytical solution is not possible. Non-linear ODEs are examples of equations with no analytical solution. Numerical methods can also be easily implemented as computer programs. Automating the numerical method with a programming language reduces the calculation time and the number of input and calculation errors.

There are several numerical methods available for solving ODEs. Each has its own advantages and disadvantages. The methods vary in accuracy, time complexity, and storage complexity. Advanced methods may even include the ability to control approximation errors.

This section is an introduction to two fundamental methods for integrating ODEs. The methods are presented in order of increasing accuracy and complexity. The first method is the Euler Method and the second is the 4^{th} order Runge-Kutta Method (RK4).

Theory:

Numerical methods are useful for solving initial value problems (IVPs). IVPs are problems for which an initial condition is supplied and is used to solve for a specific solution. The Euler method and the RK4 method are iterative processes. They use the initial condition as a starting point to approximate the following point and then repeat the process until the upper limit is reached. The result of the Euler method and RK4 method is a table of points spaced equally across the x-axis and within a given interval. New points are approximated using the previous point, a slope approximation, and the length of the interval between points. Several inputs are required to begin the approximation process.

Both the Euler method and RK4 method require at least four inputs to integrate a single ODE. The first two pieces of data are the x and y values which together are the initial condition. They are represented in the form [$y(x_0) = y_0$]. The [x_0] value is also the lower limit of integration. The third piece of data is the upper limit of integration [x_n] which determines how far to integrate along the x-axis. The fourth input is the number of steps [n] to use between the limits of integration [x_0, x_n]. The number of steps will vary depending on the method used and the function being integrated. Selecting a reasonable number of steps is the biggest factor in reaching an accurate approximation. If the number of steps is too small, truncation errors will be large. However, too many steps increases round-off error.

The step size [h] is also required and is calculated by subtracting the lower limit the upper limit and dividing by the number of steps. If a specific step size is desired rather than a given number of steps, rearrange the equation to solve for n. Then use the limits of integration and desired step size to calculate the number of steps.

All data was generated using C++ numerical integrators developed for this project. The following ODE and initial condition is used for examples for the Euler method and RK4 method.

$$\frac{dy}{dx} = x\sqrt{y} \qquad \text{while} \quad y(1) = 4 \qquad \text{and} \qquad 1 \le x \le 1.5$$

The Euler Method:

The Euler Method is the most basic numerical method for solving ordinary differential equations. It's used to calculate a straight-line approximation to the next dependent variable solution point. It's derived through the following process.

$$y - y_1 = m(x - x_1)$$

Start with the equation for a line in point-slope form.

$$y - y_i = F(x_i, y_i)(x - x_i)$$

Let x_1 and y_1 be equal to x_i and y_i which are the approximations for the current iteration. Then, substitute $F(x_i, y_i)$ for m which is the derivative of the function evaluated at the current point.

$$y_{i+1} = y_i + F(x_i, y_i)(x_{i+1} - x_i)$$

Assume x and y are coordinates for a point on the graph where x is greater than x_i. Now, substitute x_{i+1} for x and substitute y_{i+1} for y. Then, rearrange the equation to solve for the next y value.

$$y_{i+1} = y_i + F(x_i, y_i)h$$

The difference between the value [x_{i+1}] and [x_i] is also known as the step-size [h]. This yields the final form for the Euler Method.

The differential equation (also the integrand) describes the slope of the equation being approximated. Evaluating the ODE at the current x value produces the slope approximation to the next point. To approximate the next y value, multiply the slope by the step size and add it to the current y value. Then, update the x value by adding the step size to it. This process is repeated until x reaches the upper limit of integration. The data set can be graphed to show the data points and the curve's trend.

Solutions for an example equation calculated using an Euler method program are shown below. The following problem used 5 steps resulting in a step size of 0.1. Table#01 contains Euler method y value approximations in the second column compared to the exact solutions in the third column. The exact values are obtained analytically using the separation of variables technique to solve the initial value problem. Figure#01is a graph containing the Euler approximations in blue and the curve showing the exact solution in red.

X	Euler Y	Exact Y
1	4	4
1.1	4.2	4.21276
1.2	4.42543	4.4521
1.3	4.67787	4.71976
1.4	4.95904	5.0176
1.5	5.27081	5.34766

Table#01: Euler approximations compared to the exact, analytical solutions.

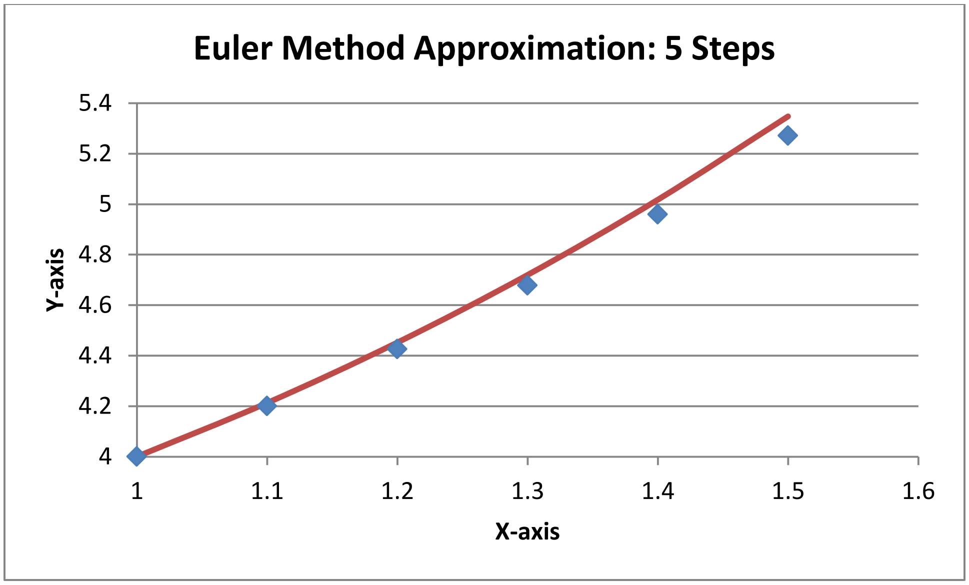


Figure #01: A plot of the data in Table#01. The diamonds are approximations from the Euler method. The curve represents the exact solution.

A small, but noticeable difference exists between the numerical solution and the exact solution. With each iteration, the difference between the approximation and analytical solution increases. The approximations for the example problem look fairly accurate because the curve is smooth and is integrated over a small interval. As the number of iterations increase, the global error for the approximation increases.

Global error and local error are classified as truncation errors. Global error refers to the total error attributed to an approximation after multiple iterations. It's equal to the summation of the local error for the iterations already completed. The local error is the error produced from a single iteration assuming the approximation was made from an exact solution. Sometimes the global error has a known maximum value. So, local truncation error tends to start large and then decrease as the summation of the local errors approaches the maximum global error.

The Euler method doesn't account for changes in the function's slope in the interval between [x_i] and [x_{i+1}]. This issue is exacerbated when working with stiff equations. Stiff equations are typically characterized by oscillations with different frequencies which cause large fluctuations in the slope.

To reduce the error for stiff equations, the Euler Method must be run using a large number of steps resulting in extremely small step sizes. For hand written solutions, using small step sizes becomes a computationally intensive and error prone process. As the number of steps increases, the total round-off error for the problem will grow. Round-off error exists because numerical methods use finite numbers to represent real numbers which can be infinite. The error builds up due to rounding within individual iterations. If the number of iterations grows too large, the approximations may diverge from the exact solution. Better methods exist for obtaining approximations for stiff equations. Some methods, such as the Runge-Kutta family, use a weighted average of slopes at various points inside the current step, while other methods use varying step-size to control error.

Euler Method approximations are poor because the method uses a single slope approximation calculated using the current coordinates. There are a couple scenarios where a predictable error pattern may emerge. For functions that are concave up, the Euler Method approximations will underestimate the true solution. This happens because the magnitude of the slope continues to increase between $[x_i]$ and $[x_{i+1}]$. The approximation at $[x_i]$ is not a good representation of the slope across the entire interval. Figure #01 shows the Euler Method's underestimation. With each step the underestimation becomes worse. Steeper slopes lead to larger underestimations. We can use the same concept to show overestimations for concave down functions.

To understand the benefits of smaller step sizes with the Euler method, look at the following comparison between approximations using 5 steps and approximations using 20 steps. The comparison is shown in Table#02 and Figure #02. It's hard to discern the difference between the approximations on a scatter plot. Instead, the table and figure show error between the approximations and the exact solution for each step size. Table#02 and Figure#02 compare the percent error for the 5 step approximations and the 20 step approximations. Notice the substantial decrease in percent error as the number of steps increase.

X	%Error 5 Steps	%Error 20 Steps
1	0.00%	0.00%
1.1	0.30%	0.08%
1.2	0.60%	0.15%
1.3	0.89%	0.23%
1.4	1.17%	0.30%
1.5	1.44%	0.37%

Table#02: Column two lists the percent error for a 5 step approximation. The third column shows the percent error for the same intervals, but was calculated using 20 steps.

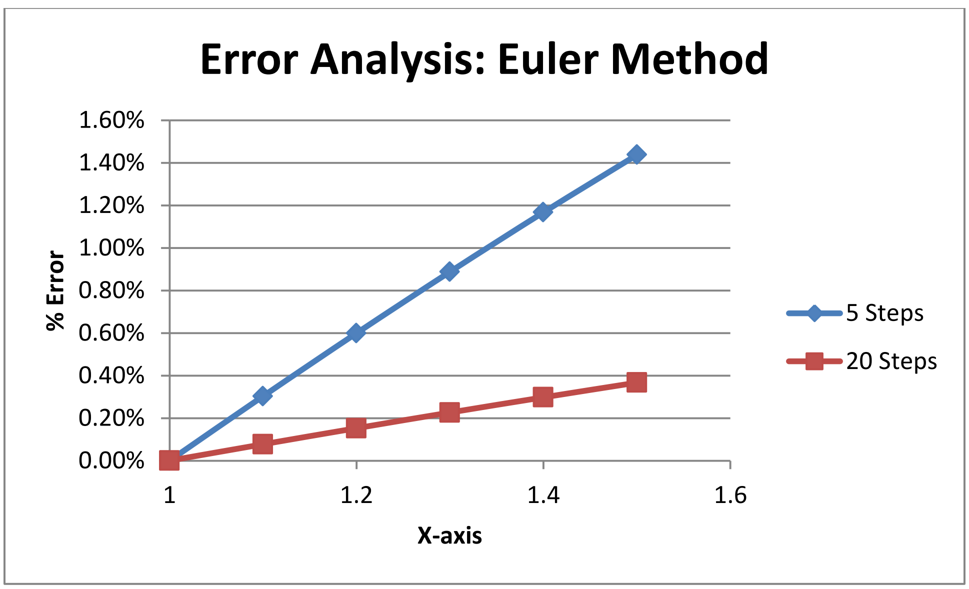


Figure #02: The curve following the diamond points shows the percent error for a 5 step approximation. The square points represent the percent errors at the same points using 20 steps for the integration. The graph shows the decrease in percent error as the number of steps increases and step size decreases.

4th Order Runge-Kutta:

The 4th order Runge-Kutta (RK4) method is the most fundamental numerical method which produces reliable approximations. The 4th order name signifies two things. The first is the number of slope approximations averaged to calculate the final slope approximation. The second is that it's typically 4 times more accurate than the Euler method.

The four slopes are given in the variables k_1-k_4, and are used to approximate the true slope. Each k value uses the preceding k value in its calculation. The first approximation k_1 is the same equation used to approximate slope for the Euler method. The second and third k values are evaluated at the current interval's midpoint $[x_i + {}^{h}/_{2}]$. The fourth k value is approximated at x_{i+1}. k_1 through k_4 are described by the following equations.

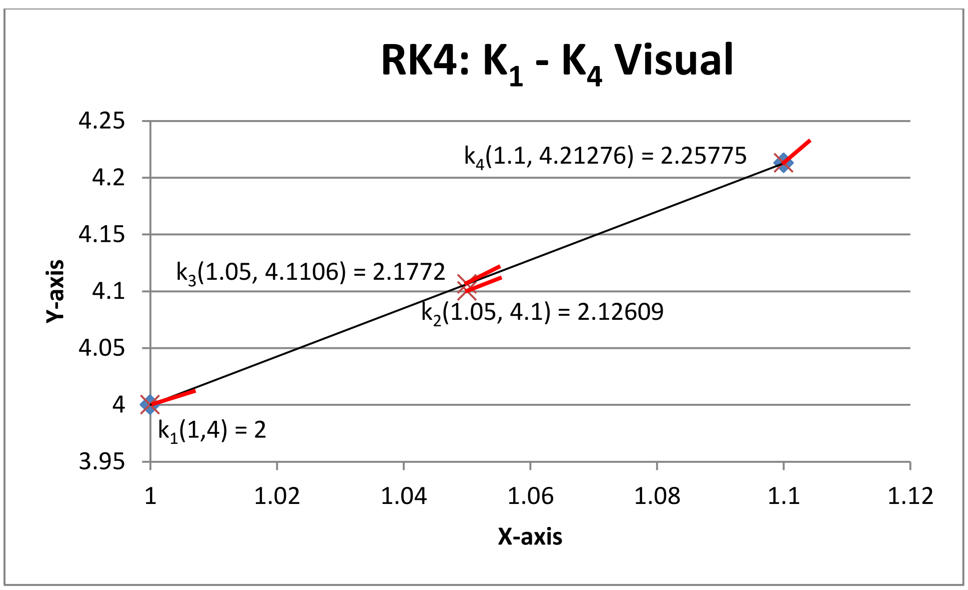
$$k_1 = f(x, y)$$

$$k_2 = f(x + \frac{h}{2}, y + \left(k_1 * \frac{h}{2}\right))$$

$$k_3 = f(x + \frac{h}{2}, y + \left(k_2 * \frac{h}{2}\right))$$

$$k_4 = f\left(x + h, y + (k_3 * h)\right)$$

The points and slopes for k_1 through k_4 are shown for the example ODE problem in Figure#03.



Figure#03: The four x points show where each of the four k values is evaluated. The short lines show the approximation to the true slope at each point. The true slope approximations are used to calculate an average slope approximation.

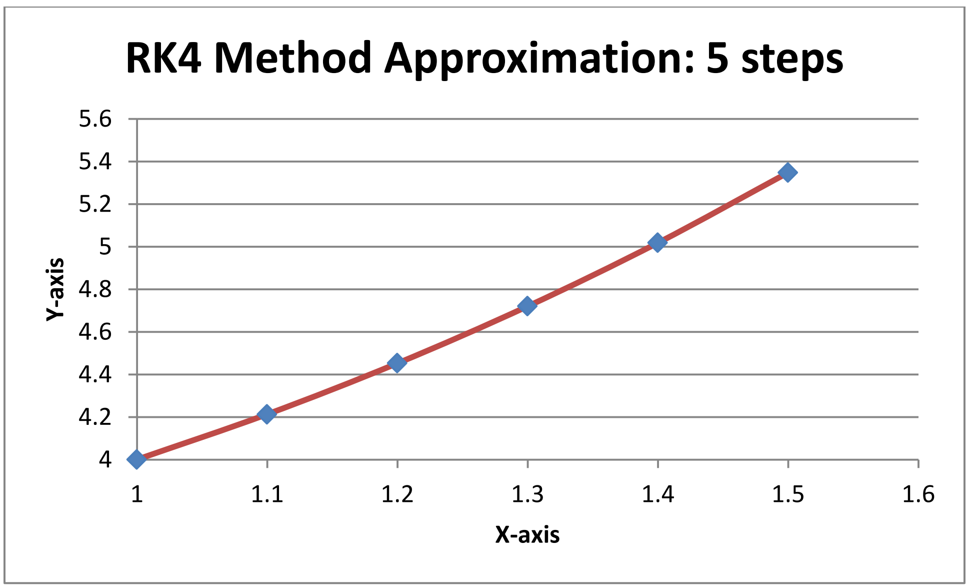
RK4 uses a weighted average of k_1 through k_4 to approximate the next point. The k_2 and k_3 approximations are given a larger weight than k_1 and k_4. These values tend to be better slope approximations because they are located at the midpoint of the interval. The slope approximation is then multiplied by the step size and added to the current y approximation. This yields the new y approximation. The following equation is the formula used to calculate new RK4 approximations.

$$y_{i+1} = y_i + (\frac{h}{6})(k_1 + 2(k_2 + k_3) + k_4)$$

Table#03 and Figure#04 show the approximate solutions for the sample ODE problem compared to the exact, analytical solutions. This example uses a step size of 0.1.

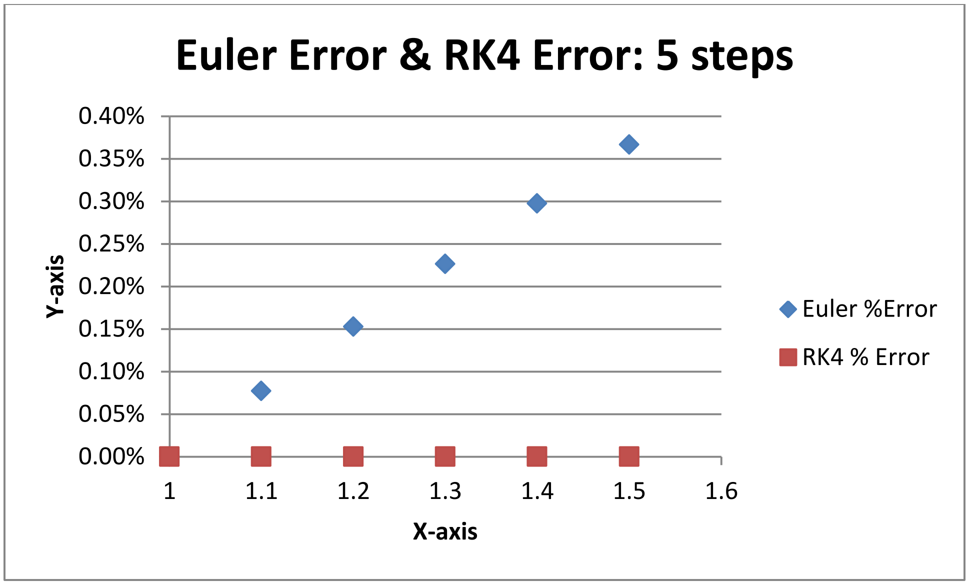
X	RK4 Y	Exact Y
1	4	4
1.1	4.212756	4.21276
1.2	4.4521	4.4521
1.3	4.719756	4.71976
1.4	5.0176	5.0176
1.5	5.347656	5.34766

Table#03: The second column lists the approximations made using the RK4 integrator. The third column shows the exact, analytical solutions for comparison.



Figure#04: The RK4 approximations are plotted as diamonds. The exact solution is represented by the curve.

Notice how much closer the RK4 approximations are to the exact solution curve than the Euler method approximations in Figure#01. The percent error for each method's approximations at each point in the interval is shown in Figure#05.



Figure#05: The diamonds represent the percent error for the Euler method compared to the true solution. The square points show the percent error for RK4 approximations. The graph shows a RK4 produces a smaller percent error for approximations compared to the Euler Method.

The percent error for the Euler method approximations grow almost linearly while the percent error for the RK4 method is never larger than 0.0001%. The percent error seems small, but the region being integrated is also very small. The curve is also fairly smooth. These conditions cannot get much more ideal for the Euler method. This comparison shows the RK4 method's superiority due to the weighted average slope approximation.

Systems of Equations:

The above examples for the Euler Method and the 4th order Runge-Kutta Method solve 1st order ODEs. The Euler Method and RK4 are capable of solving higher order ODEs, but the differential equation must be split into a system of 1st order equations. Systems of equations essentially allow the numerical integrators to gather data for more complex models.

The following is an example for a system of differential equations for a physics problem involving motion.

Position: $r(t)$

Velocity: $v(t) = \frac{dr}{dt}$

Acceleration: $a(t) = \frac{dv}{dt} = \frac{d^2r}{dt^2}$

Acceleration is the second derivative of position with respect to time, but it can be written as the first derivative of the velocity with respect to time while velocity is written as the first derivative of position. Using the system of equations, the position is approximated using a slope approximation from the velocity function. Then, because the acceleration is written as the first derivative of velocity, it can be used to approximate the slope for the velocity approximation.

When solving systems of ODEs, the number of equations depends on the original order of the differential equation. One differential equation is required for each solution. Also note, the number of initial conditions should match the number of 1^{st} order equations in the system.